Cours de traitement d’images
Corrigé d’exercices
4. Filtrage numérique des images

14.12.1999

1. ***Transformée de Fourier discrète bidimensionnelle*** :

   Soit les images `mit.tif` *et* `camman.tif` *en niveaux de gris. Calculer leur transformée de Fourier à l’aide de la fonction Matlab* `fft2` *et les afficher avec les basses fréquences au centre de l’image (utiliser* `fftshift`*).*

   (a) *Modifier la valeur des images transformées en annulant la* phase *de leurs coefficients*

   (b) Même question, en donnant le même module *à tous les coefficients*

   (c) *Créer une image à partir de la phase de* `mit` *et du module de* `camman`*.*

   (d) *Quelles conclusions tirez-vous de ces différentes manipulations ?*

   Soit `mit` l’image originale. Son chargement et son affichage à l’aide de Matlab se fait ainsi :

```
>> x = imread('mit.tif');
>> imshow(x);
```

   On procède de même pour l’image `camman`. On calcule ensuite la transformée de Fourier (complexe !) de l’image. Il est intéressant de représenter la norme de la transformée de Fourier, puisqu’elle représente l’amplitude des fréquences spatiales de l’image. En intensité, on visualise ainsi :

```
>> y=fft2(x);
>> ymod = abs(fftshift(y));
>> imagesc(log(ymod)); axis('square');
```

   L’utilisation de la fonction `imagesc` ci-dessus permet d’utiliser toute la gamme dynamique de l’image pour la visualisation. On peut vérifier que `sum(sum(x)) = y(1,1) = max(max(y))`. La fonction `fftshift` permet de recentrer y pour visualiser l’amplitude du contenu fréquentiel avec les basses fréquences au centre de l’image (domaine principal).

   La phase du contenu fréquentiel de l’image est plus difficile à appréhender. Afin de se rendre compte de l’importance de l’information contenue dans la phase, on peut visualiser l’image de phase comme suit :

```
>> z = ifft2(y./abs(y));
>> imagesc(abs(z)); axis square;
```

   Ces résultats sont donnés dans la figure 1.

   L’image de la figure 2 représente l’image de la transformée inverse de la “transformée” formée de la phase de la transformée de l’image `mit` avec le module de la transformée de l’image `camman`. Les commandes pour obtenir ce résultat sont les suivantes :

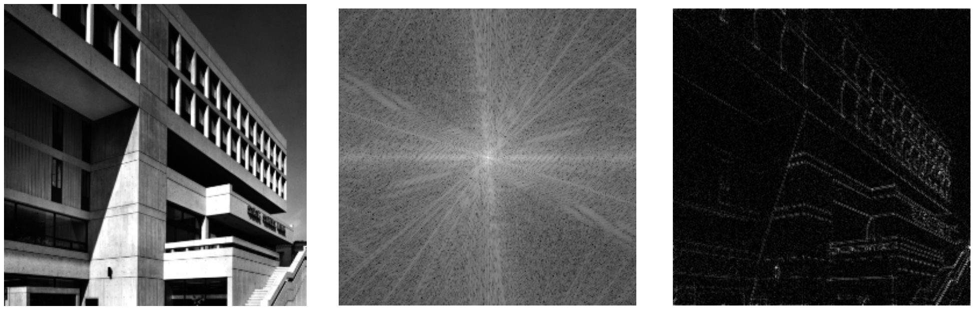

Figure 1: Image originale (à gauche), logarithme du module de la transformée de Fourier recentrée (au centre) et image de phase (à droite).

```
>> Fcamman=fft2(camman);
>> Fmit=fft2(mit);
>> Fmix=abs(Fcamman).*exp(i*angle(Fmit));
>> mix=ifft2(Fmix);
>> imagesc(abs(mix));
```

L'information portée par la phase de l'image semble nettement plus significative que celle portée par le module (cf. fig 2). En effet, la phase informe sur l'emplacement et l'orientation de tous les contours de l'image, alors que le module ne renseigne que sur "l'intensité" de ces contours.

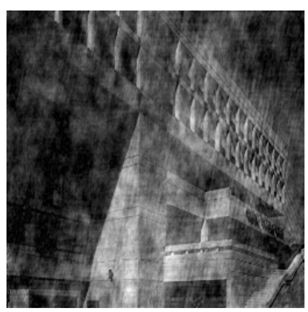

Figure 2: Importance de l'information contenue dans la phase de la transformée de Fourier

2. ***Filtrage par convolution et par transformée de Fourier*** :

   Soit un système ayant en entrée une image x de taille $M \times N$ et en sortie l'image après filtrage $y = g ** x$ avec un filtre g de taille $K \times L$.

   (a) *Calculer et visualiser y dans le cas d'un filtre $g = \texttt{ones}(K, L)$ pour $(K, L) = (1, 8)$, $(8, 1)$ et $(8, 8)$ (utiliser les fonctions `conv2` ou `filter2`).*

   (b) *Calculer y en passant par le domaine fréquentiel (à l'aide de la fonction `ifft2`).*

   (c) *Que constatez-vous ?*

   (d) *En déduire l'utilité de g en fonction de (K, L).*

   Le filtrage de x par convolution avec g est immédiat :

```
>> X = imread('camman.tif');
```

```
>> imagesc(filter2(ones(1,8),X)); axis square;
>> figure
>> imagesc(filter2(ones(8,1),X)); axis square;
>> figure
>> imagesc(filter2(ones(8,8),X)); axis square;
```

L'application de ces filtrages à x est représentée à la figure 3. Pour $(K, L) = (1, 8)$, il s'agit d'un filtre passe-bas horizontal très prononcé, résultant en un "flou horizontal". De même, pour $(K, L) = (8, 1)$, le filtrage est passe-bas vertical. Dans le cas où $(K, L) = (8, 8)$, le filtre est passe-bas isotrope.

Figure 3: Filtrage passe-bas horizontal (à gauche), vertical (au centre) et isotrope (à droite).

Si on passe par le domaine fréquentiel, le filtrage passe-bas isotrope est donné par :

```
>> TFX = fft2(X);
>> TFG = fft2(ones(8,8),256,256);
>> TFY = TFG .* TFX;
>> y = ifft2(TFY);
```

Le résultat obtenu dans les deux cas est comparé à la figure 4. Il est important de noter l'adjonction de zéros à la matrice g de manière à respecter la taille de l'image d'entrée. Les légères différences apparaissant dans les images de la figure 4 sont dues au fait que dans le domaine fréquentiel, lorsque le filtrage s'applique près des bords de l'image, l'image est répétée de part et d'autre (puisqu'on est dans le domaine de Fourier). Cet "effet de bords" est clairement visible dans le haut de l'image où les jambes du cameraman réapparaissent. Dans le cas de l'utilisation de `filter2`, des zéros sont ajoutés à l'image originale (cette technique est appelée *padding* en Anglais) lorsque le filtre s'approche des bords, créant un autre type d'effet de bords bien connu.

3. ***Filtrage d'une image par Laplacien*** :

   (a) *Appliquer l'opérateur Laplacien discret bidimensionnel vu durant la séance 3 à une image quelconque et en déduire l'utilité de ce filtrage.*

   (b) *Que se passe-t-il si on effectue la différence entre l'image originale et l'image ainsi filtrée par le Laplacien ?*

   (c) *A l'aide de la fonction* `mesh` *et des tables de couleurs* `hsv` *et* `hot`*, afficher la réponse fréquentielle du Laplacien*

La réponse impulsionnelle du Laplacien est bien connue :

$$g(k, l) = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Figure 4: Filtrage isotrope par multiplication des transformées dans le domaine fréquentiel (à gauche) et par convolution (à droite).

L'application de filtre à une image s'effectue de la même manière que pour les exercices précédents, à savoir en utilisant la fonction `filter2`. La figure 5 montre le résultat obtenu après filtrage de l'image `camman` par le Laplacien : il s'agit bien d'un filtre utile pour la détection des contours.

Si on soustrait l'image filtrée de l'image originale, on opère un "rehaussement" des contours de l'image originale.

Figure 5: Détection des contours (à gauche) et rehaussement (à droite).

La réponse fréquentielle du Laplacien se calcule à l'aide de la fonction `fft2`. On a utilisé la fonction `surfc` pour représenter cette réponse :

```
>> Fg = fft2(g,256,256);
>> surfc(abs(fftshift(Fg))); shading flat;
```

Cette réponse fréquentielle est donnée dans la figure 6.

Remarque : si on essaie de filtrer l'image via la transformée de Fourier, il faut tenir compte d'un point important concernant la représentation du filtre g. En effet, dans Matlab, les matrices ont toujours leur origine en $(1, 1)$, c'est-à-dire le premier élement de la première ligne. Il faut donc réarranger les éléments de g avant d'appliquer la transformée. Cela peut se faire en utilisant successivement les fonctions `fftshift`, `fliplr` et `flipud` pour obtenir finalement :

$$g_{new}(k, l) = \begin{pmatrix} -4 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

Il faut également penser à ajouter ensuite des zéros lors de la transformée de Fourier. Ceci est aisé grâce à la fonction `fft2`.

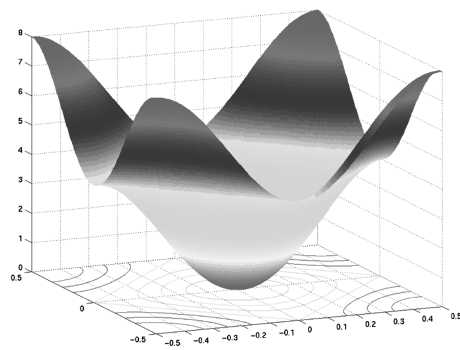

Figure 6: Réponse fréquentielle du Laplacien